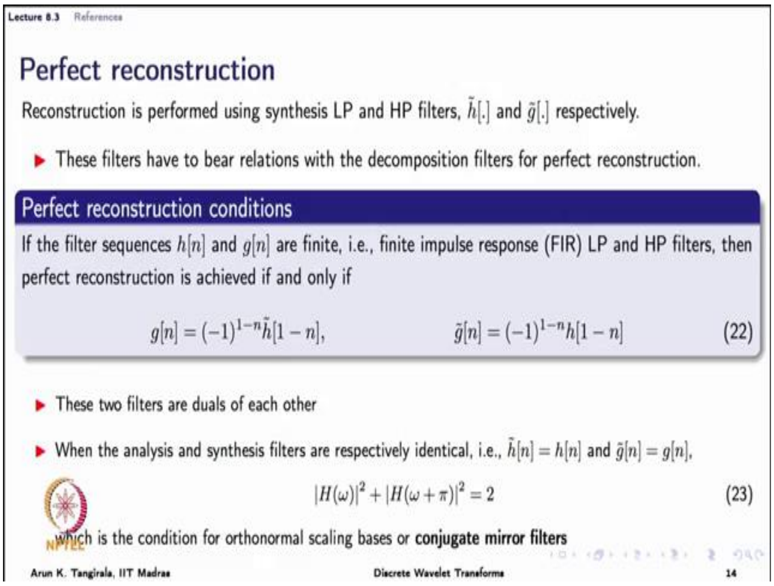
Introduction to Time Frequency Analysis and Wavelet Transforms
Prof. Arun K. Tangirala
Department of Chemical Engineering
Indian Institute Technology, Madras

Lecture - 8.3
Wavelets Filters and Fast DWT Algorithm
Part 3/3

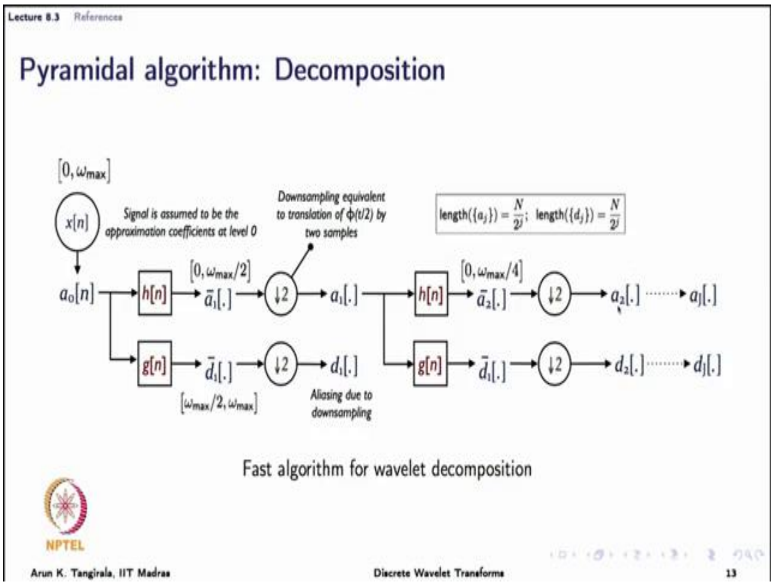
Now, when I want to reconstruct…

(Refer Slide Time: 00:15)



So, hopefully, now, you have understood the fast algorithm. I want to reconstruct. Why do I want to reconstruct? Because I want to obtain approximations or details or components of the signal in a particular frequency band.

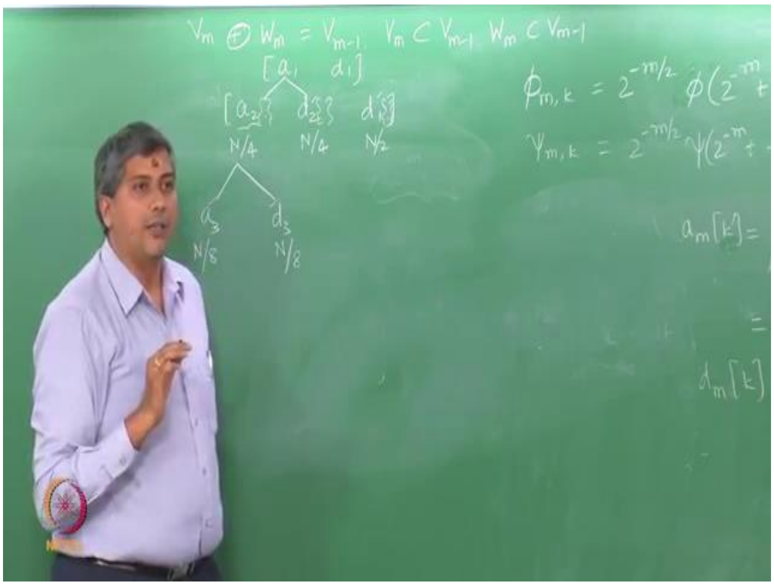
(Refer Slide Time: 00:34)



So, for examples going back to this figure; suppose I want the signal component in the range 0 to omega max over 2. And, I do not want any other signal component. Then, what I could do is I could take a 1 alone and reconstruct the corresponding part. That is what we have denoted as x hat of 1 earlier – x hat 1 of t. And, likewise, I may be interested only in the frequency range – omega max over 2 to omega max or omega max over 4 to omega max over 2 and so on. Or, maybe a certain set of components put together and some. This is the basic idea in signal estimation. When I want to do that, I need to know how to reconstruct. Now, suppose I want to recover the entire signal; I have just decomposed; I should be guaranteed that, if I put together all the coefficients, then I should be able to reconstruct perfectly.

Now, one thing that you should remember in decomposition is when I start off with a naught of n and go up to a certain level; let us say I go up to level 2; in a level 2 decomposition, what I would have at the end is a 2, d 2 and d 1. These three sets of coefficients are sufficient to completely recover x. I do not need a 1. Why? Because a 2 and d 2 have taken birth from a 1; they have the complete information for you to reconstruct a 1. So, the way these coefficients are reported by many software packages is a 2 is stacked… d 2 is stacked next to a 2 and d 1 is stacked next to d 1.

(Refer Slide Time: 02:41)



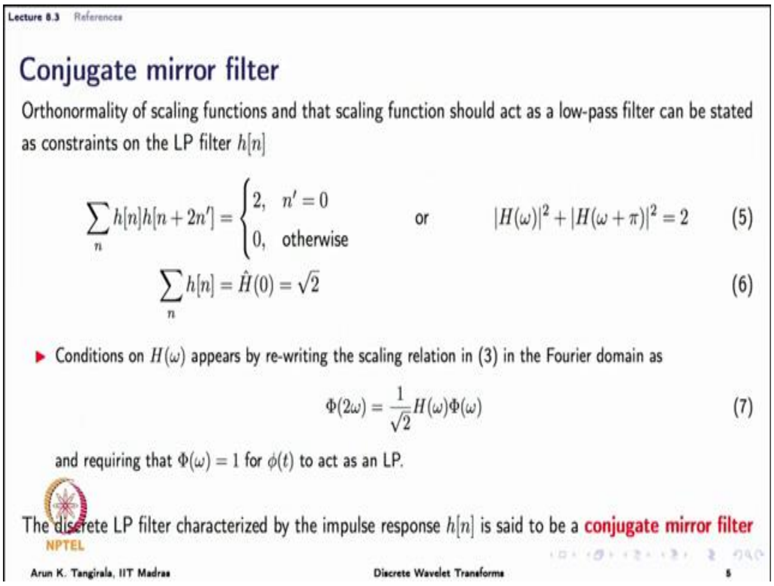
Let me just show that to you on the board. So, if I perform a two-level decomposition; then, I would have coefficients a 2, d 2 and d 1. This is how one would report. Of course, as a column vector or a row vector, that convention rest with the software package. But, what you should remember is if the signal is of length n, then a 2 – the sequence here would be of length n by 4. Essentially, n by 2 power 2 and d 2 would be of length n by 4; and, d 1 would be of length n by 2. That is a consequence of down sampling. In arriving at d 1, we have down sampled ones by factor of 2. In arriving at a 2 and d 2, a down sample twice, because a 2 and d 2 are coming from a 1 alone. Of course, at level one, I would have a 1 and d 1. But, a 2 and d 2 are taking birth from a 1. And further, if I go down; then, a 2 would be decomposed. This is called a wavelet tree – DWT tree. And, I would obtain a 3, d 3, and so on; in which case, a 3 would be of length n by 8; d 3 would be of length n by 8 and so on. So, when I report the decomposition – wavelet decomposition at a certain level; if it is a two-level, I would report a 2, d 2, d 1. If it is a three level, I would report a 3, d 3, d 2 and d 1 and so on. The total length at any level decomposition will always be equal to the length of this signal. That is the basic idea.

Now, in perfect reconstruction, what I need to be guaranteed is – let us say I look at a three-level decomposition; if I start… If I take the three-level decomposition coefficients – a 3, d 3, d 2 and d 1; and, I perform the inverse operations. So, what I have done here is I have convolved and I have down sampled. The inverse operations would also involve some kind of filtering; but, it would involve up sampling. And, there are conditions under which only you can perfectly recover this signal. These conditions are on the filters that are used in reconstructing. That is the basic idea. Whichever level I start with, the moment I am given all the necessary coefficients, I should be able to recover the signal. That is the condition of perfect reconstruction. I avoid all the theoretical discussion here. But, I will give you straight away the condition for perfect reconstruction. Assume that, this h and g are finite in length. Those are familiar with signal processing. We will know that, we are referring to FIR filters – finite impulse response filters, because h and g can be also infinite impulse response filter. There has been… We have never imposed such a restriction on h and g anywhere in our discussion until now.

But, normally, the kind of filters that we use in wavelet analysis are FIR filters. And, for FIR filters, that is, if I perform the wavelet decomposition with FIR filters, then the synthesis filters will have to bare a certain relation with the analysis filters as given by Smith and Banwell and… In fact, more generalization of this is due to Vetor Lee. And, the derivation of this… The proofs of this statement is given nicely in Mallat's book and elsewhere in literature. I recommend that you look through it if you are really interested. But, from a practical view point, what we are interested in knowing is how these synthesis and analysis filters are related. And, these expressions are given in 22. h tilde and g tilde are the synthesis filters. And, g and h as we know are the analysis filters. So, you can see that, g of n… h tilde is dependent on g; that is, the synthesis low-pass filter is dependent on the high-pass filter. And, the synthesis high-pass filter is related to the analysis low-pass filter; so, since this low pass is related to analysis high pass and synthesis high pass is related to analysis low pass.

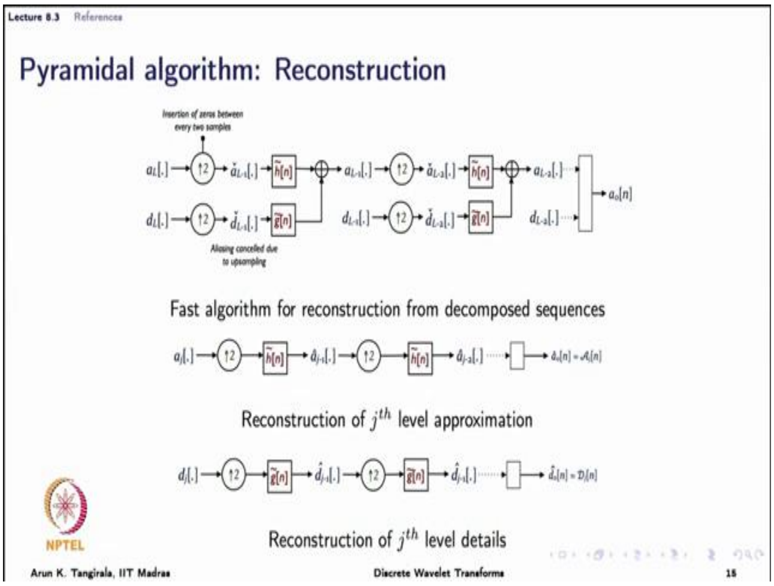
Although we have been using these terms: synthesis and analysis, these two sets of filters play dual roles; which means I could consider h tilde and g tilde as analysis filters. Then, h and g become synthesis filters; or, I can consider h and g as analysis; and, g tilde and h tilde become synthesis filters. So, their roles are reversible; pretty much like what we said in frame theory. We talked about frames and then we talked about dual frames and we said dual frames and frames can play reverse roles. I can call one frame; fix one as a frame and call the other as a dual frame. When the analysis and synthesis filters are respectively identical. So, this condition is for general synthesis and analysis filter. But, if I require that, g tilde equals g and h tilde equals h, then you can show that, these two conditions can be written nicely in the fourier domain, that is, in terms of the frequency response functions as a single condition given in equation 23. So, now, I do not have h tilde and g tilde and h and g separately. The same h and g that I use for analysis will be used for h and g for synthesis as well. If I do that, what condition should the filter satisfy? Now, we know everything is tied only to one decision, which is that of low-pass filter as long as that low pass filter satisfies this condition. Now, this is a condition that we have seen earlier.

(Refer Slide Time: 09:04)



If you recall, this is a condition that we had said is required for orthonormality of these scaling functions. Automatically, h satisfies this. So, you do not have to really do anything. The moment I choose h that corresponds to a scaling function and essentially satisfying equation 5, and I design g according to this relation 14; then, I have the analysis and synthesis filters already. This 14 comes from the fact if you recall that, I want orthonormal wavelet basis. So, to summarize, I choose h as a low-pass filter satisfying equations 5 and 6. That is the conditions of orthonormality and low-pass filter. Then, I choose g such that I get orthonormal wavelet bases according to equation 14. Then, I can use both these filters for analysis as well as synthesis. And, that is the reason they are also known as conjugate mirror filters. So, h and g of course, are known as conjugate mirror filters mostly because of equation 14 that we have seen. But, these conjugate mirror filters also provide perfect reconstruction.

(Refer Slide Time: 10:19)

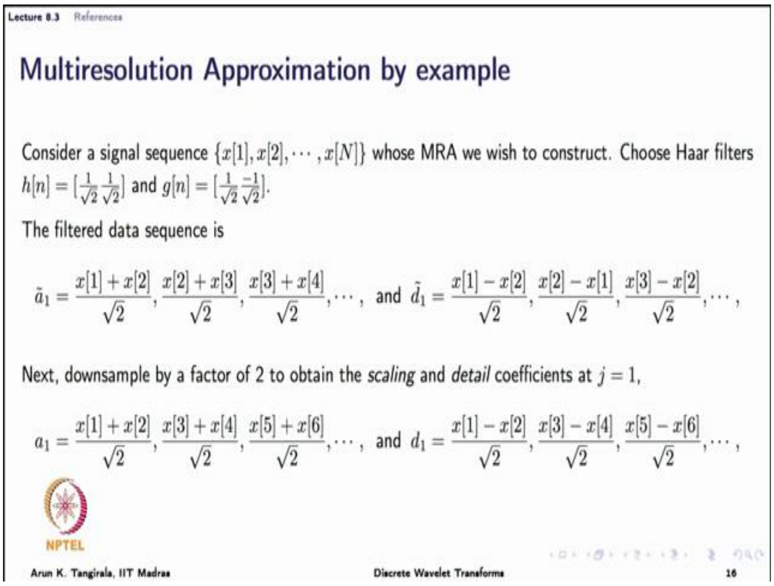


And now, we have the reconstruction algorithm at the top. So, there are three figures here. At the top, we have the perfect reconstruction algorithm, where you read essentially from here left to right. What I am generally given as you have seen earlier is an approximation coefficient and all the detailed coefficients at that scale and finer scales. So, in this algorithm here, I know a 2 for example, or this a – at the staring a here; and then, the d's at this scale and then the d at the finer scale, and the d at finer scales and so on. So, if I assume that I performed a level l decomposition, then I have a l, d l, d l minus 1, d l minus 2 up to d 1. So, I start with a l and d l up sample. This is intuitively to counter the down-sampling operation that I performed. But, theoretically, you can say that, this is to cancel the aliasing. We have not talked about that; but, the down sampling introduces some kind of aliasing. And, in the reconstruction, you have to cancel the aliasing. So, the up sampling cancels that aliasing. So, I obtain the up sampled sequences of a l and d l; convolve them with the synthesis filters. As we have just discussed, I can choose the synthesis and analysis filters to identical; add them up; I recover a l minus 1. And then, I have d l minus 1 with me because that is already given to me. Again I up sample them; reconstruct, get a l minus 2 and so on until I reach a naught, which is that a recover this sequence.

Now, what I can do is as I mentioned earlier, I can reconstruct a particular approximation – approximation in a particular frequency band; which is that, I can take a l alone and up sample; and then, that will give me a l minus 1; that is, up sample filter giving me a l minus 1; again up sample filter and so on until I reach the maximum, that is, the finest scale. When I perform this operation only on the approximation coefficient in a particular band or coefficients in a particular band, what I recover is the approximation corresponding to that frequency band; or, it is the approximation that I have corresponding to that scale. But, that is what it is essentially big A subscript l. And, I can apply this as well to the detailed coefficients; take the detailed coefficients in any band, up sample; or, you can start with d l up sample filter, use the high-pass ((Refer Slide Time: 13:30)) filter; obtain d l minus 1 hat and so on.

Now, I use hats here because ideally to produced a l minus 1, I need both a l and d l. But, here what I am doing is I am throwing away the dl; or, in other words, you can say I am zeroing out the details at level l and only retaining a l. Therefore, I do not get a l minus 1; I get a hat l minus 1. I only get an approximation of a l minus 1. Likewise here ideally, I can take dl minus 1 from what is known to me. But, if I do not want that, then I can approximately construct that from dl and so on with that. So, this is the basic idea in signal estimation. If I want to estimate… If I know that, the main component of the signal is in a particular frequency band, I zero out all the other coefficients – coefficients in all the other bands and then simply perform the reconstruction.

(Refer Slide Time: 14:34)



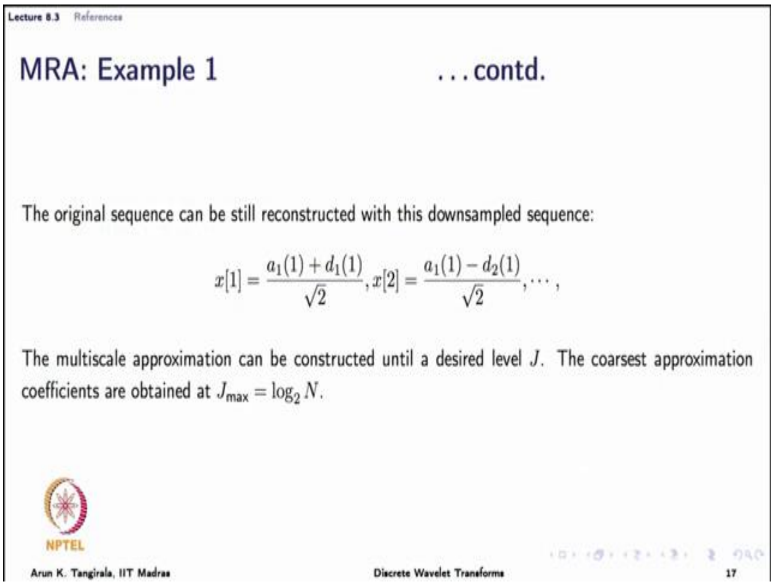
So, let us look at two examples to conclude the lecture. The first example is more of symbolic; and, the second example is more graphical and numerical. So, let us take a signal sequence here of length n. And, let us say I want to really perform the DWT or MRA. Let us choose the HAAR filters. The HAAR filters are given as 1 over root 2 1 over root 2. That is a low-pass filter. And, g of n is the high-pass filter – 1 over root 2 minus 1 over root 2. In fact, you can quickly check that, when I sum up the coefficients here in h of n, what do I get? I get root 2. So, it satisfies the condition that I had for low-pass filtering. And in fact, you should check that, if I take the fourier transform of this – h of n, it satisfies the condition, which is mod h of omega square plus mod of h of omega plus pi square is 2. So, that exercise I leave it to you. And, you can also see that, the sum of the coefficients of g is 0, which is a requirement for g to be high-pass filter. Remember g of omega at omega equal 0, is nothing but the sum of the time domain sequence. So, therefore, this satisfies all the conditions for generating MRA for computing DWT using orthonormal wavelets and scaling functions.

So, what I am trying to do here is I am showing you what kind of results you would get when you walk through the decomposition algorithm at the first step. So, at the first step, what you do is you would convolve, you would take the sequence and perform filtering. Here the filter is 1 over root 2 1 over root 2. When you perform the convolution, it is essentially taking averages of two successive points, because convolution would yield that. It is an FIR filter. So, it is performing averaging, but with a factor of root 2 here. That is the a 1 tilde that you would get. And, this is a d 1 tilde, which will have successive differences. One – you have averages of two successive points; and, the other containing differences of two successive points.

Now, down sample by a factor of 2; that is, I am going to throw away every other value in a 1 tilde. So, I am going to through away x 2 plus x 3 by root 2. I am going to throw away x 2 minus x 1 by root 2 and so on. And, when I do that, I have a 1 and d 1. Now, quickly what you should notice is; by throwing away a factor of 2, I have not lost any information at all. What do I mean by this? I can start off with a 1 and d 1 and still perfectly recover x 1, x 2, x 3 and so on. How do I do that? Pick the first value of a 1; x 1 plus x 2 over root 2; pick the first value of d 1 – x 1 minus x 2 over root 2. And then, do what? Simply perform two operations. Take the average and what would you get? You will get root 2 of x 1. If you average the first values in a 1 and d 1; simply add them up; you do not even need to take the average; simply add them up; you will get root 2 x 1 divided by root 2. What have we done? In fact, all I need to do is multiply the first coefficient with root 2 in a 1 and multiply the first coefficient of d 1 with root 2; add them up; I will get x 1.

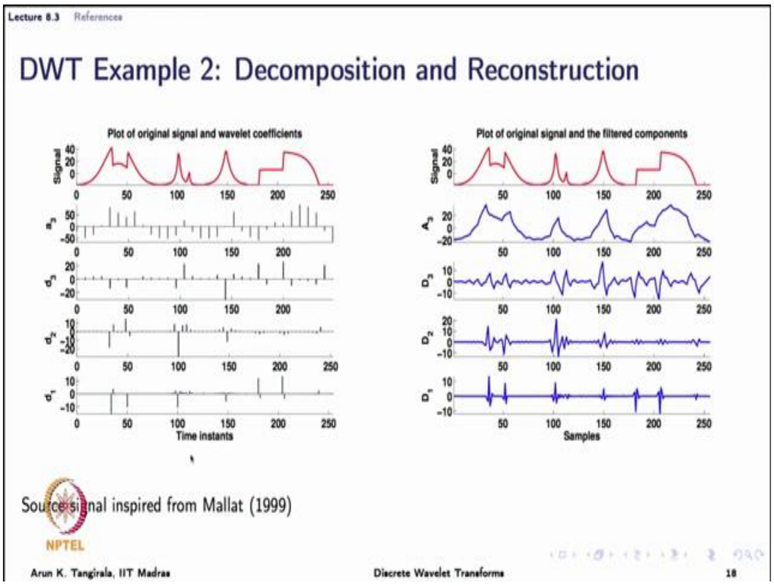
What have I done here? I have convolved a 1 and d 1 with the… In fact, ideally, I should be doing up sampling; but, I am just saying that, I have averaged a 1 and d 1 with h itself; that means, synthesis filter is also the same. h has coefficients 1 over root 2, 1 over root 2. I take the same coefficient and apply them to a 1 and d 1. Ideally, I should be up sampling also; but, I am not showing you that operation. So, this is how you can perfectly reconstruct the signal. So, this kind of tells you what you can do.

(Refer Slide Time: 18:50)



As I said here, you can recover x 1, x 2 in this fashion and so on.

(Refer Slide Time: 18:58)



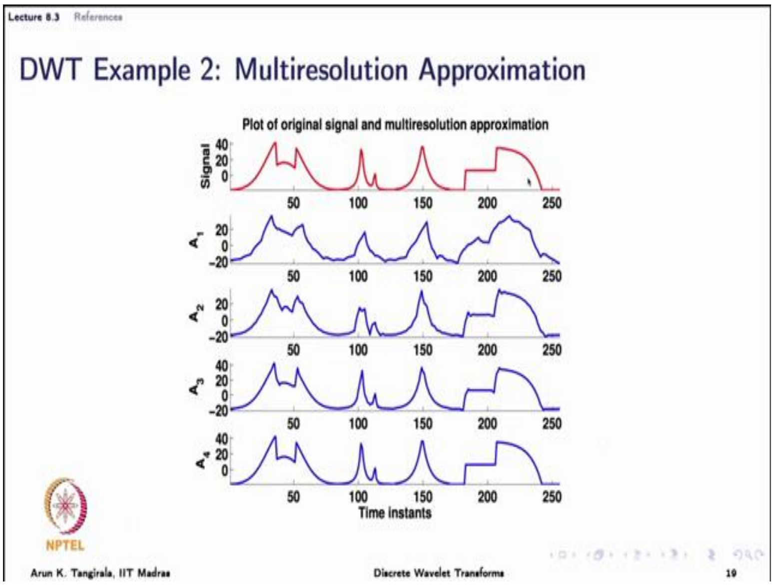
Now, let me show you the second example here on a polynomial that is used by Mallat in his book. And, this signal can also be generated using wave lab. In the next lecture, I will show you how to compute this in decomposition and the reconstructions in mat lab. For now, let us understand this decomposition. What I have taken is I have taken this piece-wise regular polynomial or signal and performed a three-leveled decomposition. When I perform three-level decomposition, I have four sets of coefficients: a 3, d 3, d 2 and d 1. And, this plot clearly shows the differing time resolutions. The signal although being shown continuous is actually a sampled sequence available; let us say every sampling interval is 1; then, the sampling interval for d 1; or, the time resolution for d 1 will be 2. For d 2, it would be 4; for d 3 and a 3, it would be 8.

So, as you can see here, I have 256 points. Here because the sampling interval is 1, the time duration is still 256; but, I have 32 points in a 3. And, I have 32 points in d 3; 64 points in d 2; and, 128 coefficients in d 1. Put together, the total length is still 256. So, if you are looking at signal compression, for example, you would see clearly that, the value of a 3 is quite high compared to the values of d 1 or d 2. So, what I could do is – in signal compression, I would say I can store only a 3 and d 3 and I can discard d 1 and d 2. Then, I would achieve a compression factor of 4, because 32 and 32 – 34 coefficients I would store and throw away the other points that I have here, which is 256 minus 64 – 192. So, I have… I would have achieved a compression factor of 4. Of course, it would be a lossy compression. But, that is the basic idea in the signal compression using wavelets. That is a crude way of compressing; but, that is the basic idea. The refinements that you can perform is – apply a certain threshold; only store significant coefficients; you do not need to discard an entire level, set of coefficients at a certain level; you can apply a threshold at every level and only store the significant coefficients at each level and so on. Those are all the refinements. But, that is a basic idea.

Now, when you look at the right-hand side plot, they are showing the reconstructions corresponding to the coefficients here. So, what we have done here; we have taken a 3 and run it through this reconstruction algorithm here. I have taken only a 3, up sampled, filtered; again, up sampled, filtered and so on. And, I would get this big A 3. And then, I have the big D 3 corresponding to the small d 3; that is, if I were to use only d 3 for reconstruction – the small d 3, I would recover the big D 3. And, an important point to keep in mind is the big A's and D's here – the reconstructed components are all at the same time resolution as the original signal. So, you have come back in time fully in at the same scale at which the signal is available.

What is a difference between – then the signal and these components? They contain different frequency components; that is a point. And, as you can see the d 1 for example here – the reconstructed component in corresponding to d 1 contains only the high frequency blurbs naturally so, because in constructing this big D 1, I have only considered the small d 1, which is in the highest frequency band – omega max over 2 to omega max. And, the high frequency bands will contain all the abrupt changes that you see in the signal. The low frequency ones are slowly changing ones would be contained in the lowest frequency band, that is, in small a 3, or the component of the signal in big A 3. You can see the big A 3 is a nicely smoothened version – smooth version of the signal itself.

(Refer Slide Time: 23:31)



Now, finally, MRA; in fact, this is not exactly MRA; but, in some sense, yes, because… And, in fact, I apologized for the ordering here; the ordering is not right here. What is read as a 1 here should be a 3; and what you see as a 3 should be a 1. So, if I take a 3, then it is the approximation of the signal at scale 2 power 3 or at level 3. When I add d 3 to a 3, then I would get the next level approximation. And then, when I add d 2 and d 3 even, then I would get the next level approximation, which I see here. And, when I add all of these, that is, a 3 plus d 3 plus d 2 plus d 1; then, I would get the signal itself. This is possible because I have used perfect reconstruction filters. That is what is a condition of perfect reconstruction. I take up all the reconstructed components in the individual bands; add them up; I will recover the signal. So, as I said, the numbering here is wrong. This should have been a 3 and then a 2, a 1, a naught. Why the last signal should be called as a naught is because you get the signal back itself. So, that is the small error. And, we will take note of that when I put up the slide for you; that is it.

(Refer Slide Time: 25:13)

So, this has been a lengthy lecture of course. But, one of the key lectures in DWT, which tells you how the DWT is computed; more importantly, how the wavelet filters are designed staring with the scaling functions. Remember we said in CWT that, in CWT, the scaling functions are derived from wavelets; whereas, in DWT, we derived the wavelet from the scaling functions and we generated in particular orthonormal wavelet basis. There exists another route, where one can design biorthogonal wavelet basis; that we will briefly talk about it in the closing lecture on this unit. And, what we have also learnt is a pyramidal algorithm and the recursive relations responsible for that fast algorithm. And then, went through a couple of examples on decomposition and reconstruction understanding the results from a DWT analysis. So, hopefully, you learnt something valuable from this lecture. In the next lecture, I will show you how to carry out all of these in mat lab. Then, we will talk about the choice of wavelets; and, how wavelets are synthesized from these filters themselves. So, that is about it. See you in the next lecture.